Two Kinds of Uncertainties

There are two kinds of uncertainties in the tracking:

full extent errors:

The physical extent of a cluster useful in pattern recognition for voting up a particular piece of phase space

position error:

the prediction of where a track went through the cluster, useful for fitting a track model to a set of clusters

Historically the tracking code developed by Alan had only full extent errors. Of course the fitter didn't like this, so Alan rescaled by sqrt(12) where needed to make a fit. This is fine for the individual struck pixels we were using at the time where this relationship is just that simple, but for ADC-weighted clusters like in the TPC there isn't a simple relationship between the two.

I've started forking the errors so that the entire code base has access to both senses of the uncertainties.

What I did

I swapped the uncertainty usage from a 3d uncertainty:
ex, ey, ez
The old code rescaled the uncertainties between a full extent error and a position uncertainty. I fixed the rescaling but some work is still needed since the TPC won't have a simple sqrt(12) relationship between the two more changes are needed.

To a covariance for the size uncertainty (this is where the tracking code gets its uncertainty right now):
(sxx,sxy,sxz)
(syx,syy,syz)
(sz,x,szy,szz)
and a separate (currently unused) covariance for position uncertainty:
(exx,exy,exz)
(eyx,eyy,eyz)
(ez,x,ezy,ezz)

Both senses are now stored throughout the code base and are never changed as the code runs.

TPC Cluster Errors To Do

(1) Create cluster-by-cluster estimates for the covariance.

SIZE : full extent uncertainty used for pattern recognition voting

ERR: position uncertainty should be used for fitting

Currently the TPC clusters have constant uncertainties:

PHG4TPCClusterizer.C

```
245
246           if ((layer > 2) && (e < energy_cut)) {
247             continue;
248           }
249
250           SvtxCluster_v1 clus;
251           clus.set_layer(layer);
252           clus.set_e(e);
253           double radius = geo->get_radius();
254           clus.set_position(0, radius * cos(phi));
255           clus.set_position(1, radius * sin(phi));
256           clus.set_position(2, z);
257
258           clus.insert_hit(cellids[zbin * nphibins + phibin]);
259
260           float invsqrt12 = 1.0/sqrt(12.);
261
262           TMatrixF DIM(3,3);
263           DIM[0][0] = 0.0;//pow(0.0*0.5*thickness,2);
264           DIM[0][1] = 0.0;
265           DIM[0][2] = 0.0;
266           DIM[1][0] = 0.0;
267           DIM[1][1] = pow(0.5*0.011,2);
268           DIM[1][2] = 0.0;
269           DIM[2][0] = 0.0;
270           DIM[2][1] = 0.0;
271           DIM[2][2] = pow(0.5*0.03,2);
272
```

Uncertainty Usage To Do

(2) Currently the code is only using the SIZE uncertainty, some calls should be swapped to the ERR covariance, which will allow a more sophisticated estimation of the position uncertainty than a simple sqrt(12) factor to be used. Examples:

```
373     for (unsigned int i = 0; i < hits_vec[zoomleve]->size(); i++) {
374       x_a[hit_counter] = (*(hits_vec[zoomleve]))[i].get_x();
375       y_a[hit_counter] = (*(hits_vec[zoomleve]))[i].get_y();
376       z_a[hit_counter] = (*(hits_vec[zoomleve]))[i].get_z();
377
378       dz_a[hit_counter] = (2.0*sqrt((*(hits_vec[zoomleve]))[i].get_size(2,2)));
379       four_hits[hit_counter] = (*(hits_vec[zoomleve]))[i];
380
381       hit_counter++;
```

Inside the voting:
No sqrt(12) scale => full extent error under use

Keep get_size() calls.

```
1353          dx1_a[hit_counter] = 0.5*sqrt(12.0)*sqrt(layer_sorted[0][i].get_size(0,0));
1354          dy1_a[hit_counter] = 0.5*sqrt(12.0)*sqrt(layer_sorted[0][i].get_size(1,1));
1355          dz1_a[hit_counter] = 0.5*sqrt(12.0)*sqrt(layer_sorted[0][i].get_size(2,2));
1356
1357          x2_a[hit_counter] = layer_sorted[1][j].get_x();
1358          y2_a[hit_counter] = layer_sorted[1][j].get_y();
1359          z2_a[hit_counter] = layer_sorted[1][j].get_z();
1360
1361          dx2_a[hit_counter] = 0.5*sqrt(12.0)*sqrt(layer_sorted[1][j].get_size(0,0));
1362          dy2_a[hit_counter] = 0.5*sqrt(12.0)*sqrt(layer_sorted[1][j].get_size(1,1));
1363          dz2_a[hit_counter] = 0.5*sqrt(12.0)*sqrt(layer_sorted[1][j].get_size(2,2));
1364
```

Inside the fitting:
Here the scale factor is taking the get_size() to a position uncertainty.

Replace with get_error() calls and no sqrt(12.) factor.

Iterative fitter in TPC version To Do

(3) The TPC version make multiple fits iteratively shrinking the error bar from the full extent to the sqrt(12) position uncertainty. I'm not sure this is the best thing to do for the TPC.

```
532  float sPHENIXTrackerTPC::fitTrack(SimpleTrack3D& track,
533                                    vector<float>& chi2_hit,
534                                    float scale) {
535
536    chi2_hit.clear();
537    vector<float> xyres;
538    vector<float> xyres_inv;
539    vector<float> zres;
540    vector<float> zres_inv;
541    for (unsigned int i = 0; i < track.hits.size(); i++) {
542
543      float ex = (2.0*sqrt(track.hits[i].get_size(0,0))) * scale;
544      float ey = (2.0*sqrt(track.hits[i].get_size(1,1))) * scale;
545      float ez = (2.0*sqrt(track.hits[i].get_size(2,2))) * scale;
```

inside the fit, "scale" changes between iterations

size uncertainty being used

My general philosophy is that the fitting code should use the position uncertainty in the get_error() call and the voting code should use the full extent uncertainty in the get_size().